CSE 559A: Computer Vision

[credit: danjodon.deviantart.com]

Fall 2017: T-R: 11:30-1pm @ Lopata 101

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Staff: Abby Stylianou (abby@wustl.edu), Jarett Gross (jarett@wustl.edu)

http://www.cse.wustl.edu/~ayan/courses/cse559a/

Sep 14, 2017

ADMINISTRIVIA

- Homework posted (and updated!). Make sure you have `pset1V2.zip`.
- Recitation will be NEXT Friday (9/22).
- Regular office hours tomorrow (in J420).

CRASH COURSE ON OPTIMIZATION

- Let x be a scalar.
- $f(x; \theta)$ is some function of x, and some other parameters θ.
- $\min_x f(x; \theta)$ is the smallest value that f can take ...
  - For some fixed values of θ
  - By searching over all possible values of x
  - Is a function of θ
  - But not of x

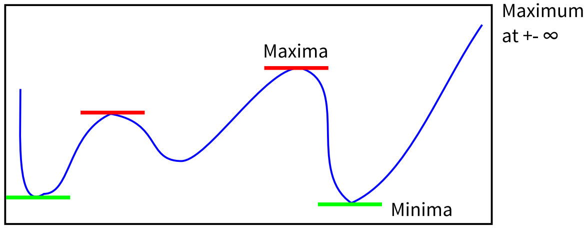
$$f(x; a, b, c) = a(x - b)^2 + c$$

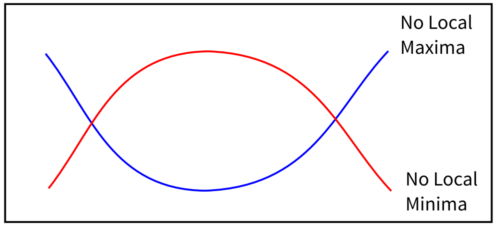
$$\min_x \quad f(x; a, b, c) = a + c$$

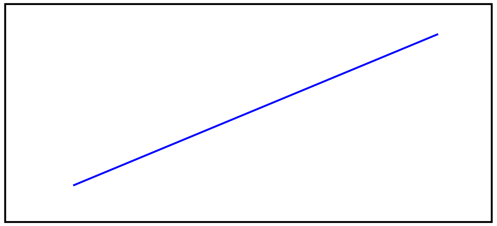
CRASH COURSE ON OPTIMIZATION

- $\arg\min_x \; f(x;\theta)$ is the value of x for which f attains its minimum value.
- Same deal for max and arg max. $\max f = -(\min(-f))$.
- How do we find x ?
- If $\frac{\partial f(x;\theta)}{\partial x} = 0$ at $x = x'$, then x' is an extremum.
    - i.e., *local* minimum or local maximum.
    - Can find which by checking second derivative.

$$\text{Minimum: } \frac{\partial^2 f(x;\theta)}{\partial x^2} > 0; \qquad \text{Maximum: } \frac{\partial^2 f(x;\theta)}{\partial x^2} < 0$$







CRASH COURSE ON OPTIMIZATION

- $f(x; a, b, c) = ax^2 + bx + c$
- Only one minima or maxima at $-b/2a$
- Can see it also by rewriting as $a\left(x - \frac{-b}{2a}\right)^2 + c - \frac{b^2}{4a}$
- Minimum if $a > 0$, Maximum if $a < 0$

CRASH COURSE ON OPTIMIZATION

- Minimization over multiple variables

$$\arg \min_{x_1, x_2, x_3} f(x_1, x_2, x_3; \theta)$$

$$\arg \min_{x} f(x; \theta), \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

- Note that output of f, which you are minimizing, is still scalar valued (a single number).

CRASH COURSE ON OPTIMIZATION

- Generalization of derivative: gradient

$$\nabla_x f(x;\theta) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix}$$

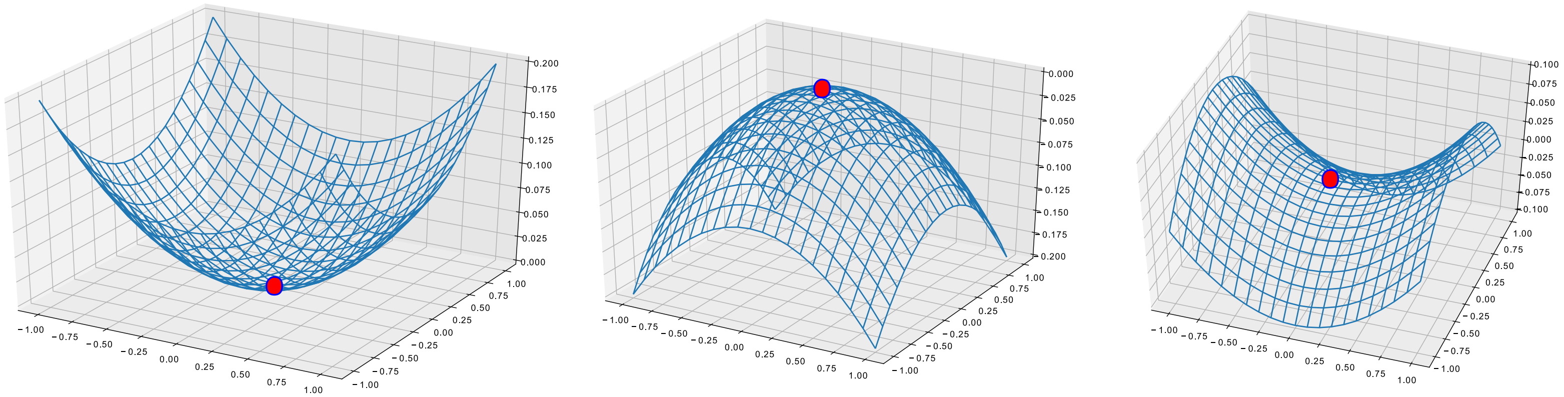
- Also a vector of the same dimensions as x

$$\frac{\partial f}{\partial(\alpha x_1 + \beta x_2 + \gamma x_3)} = \left\langle \nabla_x f, \quad [\alpha, \beta, \gamma]^T \right\rangle$$

- Derived by chain rule
- Tells us about gradient in any direction.
- $y = Ax \quad \Rightarrow \quad (\nabla_y f) = A\,(\nabla_x f)$
- If we say $(\nabla_x f) = 0$ at x, that means every element of the gradient vector is 0.
- And so, the derivative along all "directions" is 0. Then x is an extremum of f.

CRASH COURSE ON OPTIMIZATION

- Identities
  - $\nabla_x x^T Q x = (Q + Q^T)x = 2Qx$ (if Q is symmetric)
  - $\nabla_x x^T v = \nabla_x v^T x = v$
- Minima or Maxima or …

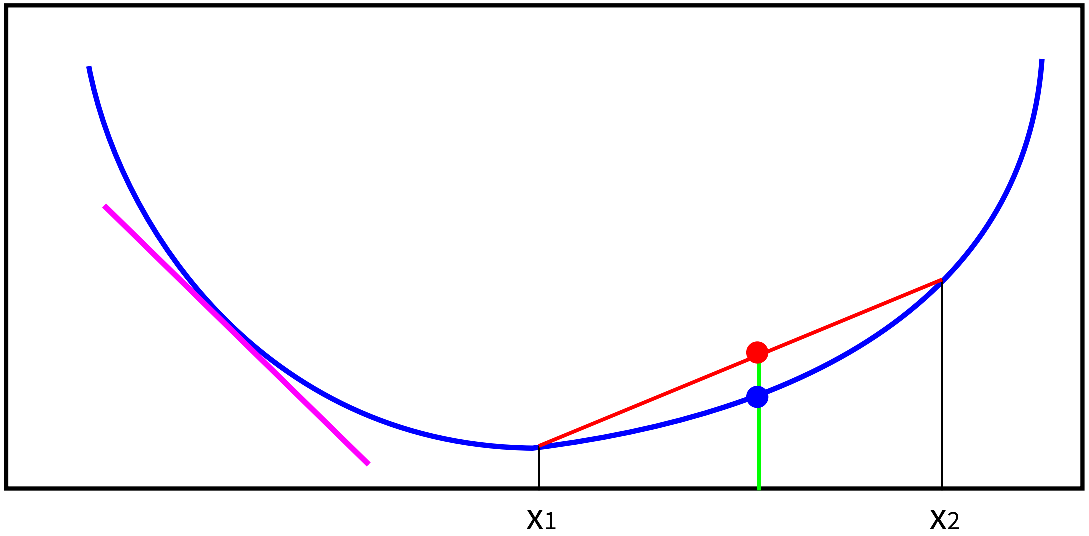


- Minimum, maximum, saddle point: things become quickly complicated in high dimensions.
- Formally, you show Hessian is positive definite: $\nabla_x (\nabla_x f)^T)$

CRASH COURSE ON OPTIMIZATION

- $f(x;\theta)$ is a strictly convex function of x, if:

$$\frac{f(x_1;\theta)+f(x_2;\theta)}{2} < f\left(\frac{x_1+x_2}{2};\theta\right), \quad \forall x_1, x_2$$



- Then f has only one local extremum. It is a local minimum, and this is the global minimum.

CRASH COURSE ON OPTIMIZATION

- Back to our setting:

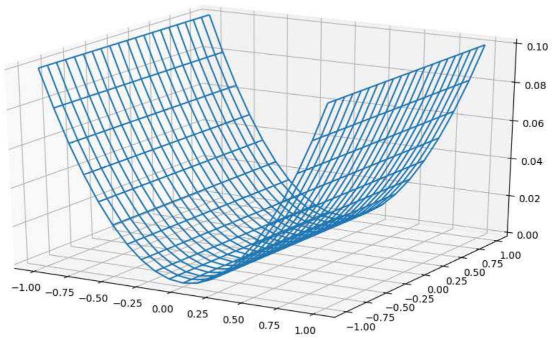
$$f(x; Q, b, c) = x^T\, Q\, x \;-\; 2b^T\, x \;+\; c$$

- Q is a symmetric positive-definite matrix.
- Multi-variable Quadratic form.
- This is convex. Single extremum which is a minimum.
- Consider eigen-decomposition of $Q = V\Lambda V^T$.
  - Columns of V are eigen-vectors. V is unitary.
  - Λ is diagonal, with eigen-values. All eigenvalues positive.
- $Q = V\Lambda V^T, \;\; x^T\, Q\, x = (Vx)^T \Lambda (Vx) = \sum_i \lambda_i (Vx)_i^2$
- Sum of quadratic terms with all coefficients (λ_i) positive

CRASH COURSE ON OPTIMIZATION

- Back to our setting:

$$f(x; Q, b, c) = x^T \; Q \, x \;\; - \;\; 2b^T \, x \;\; + \;\; c$$



Positive "semi" definite (Eigenvalues are non-negative)

CRASH COURSE ON OPTIMIZATION

- Back to our setting:

$$f(x; Q, b, c) = x^T \; Q \; x \;\; - \;\; 2b^T \; x \;\; + \;\; c$$

- Asssume Q is positive definite:

$$\nabla_x f = 0 \rightarrow 2Qx - 2b = 0 \rightarrow Qx = b$$

- $x = Q^{-1} b$

CRASH COURSE ON OPTIMIZATION

General note on computing $Q^{-1}b$

- Never compute Q^{-1}, and then multiply by b.
  - Numerically unstable, more expensive.
- Call `scipy.linalg.solve`:
  - Cholesky / LDL Decomposition: $Q = L\,D\,L^T$
  - Always exists for a positive definite matrix. L is lower triangular.
  - Solve $Qx = b \to LDL^T x = b \to Ly = b, L^T x = D^{-1}y$

$$\begin{bmatrix} a & 0 & 0 & 0 & \dots \\ q & c & 0 & 0 & \dots \\ d & e & f & 0 & \dots \\ & & \vdots & & \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \end{bmatrix}$$

DENOISING

$$X = \arg\min_X \frac{1}{2\sigma^2}\|Y - X\|^2 + R(X)$$

$$X = \arg\min_X X^T QX - 2b^T X + c$$

- $R(X) = \lambda \sum_n (x[n] - 0.5)^2 = \lambda\|X - 0.5\|^2$
- $Q = \frac{1}{2\sigma^2} I + \lambda I$
- Q is therefore diagonal.
- Q^{-1} involves inverting elements along diagonal.
- Simple to compute $Q^{-1} b$.
    - Independent operation on each pixel / element of b.

DENOISING

$$X = \arg\min_X \frac{1}{2\sigma^2}\|Y - X\|^2 + R(X)$$

$$X = \arg\min_X X^T QX - 2b^T X + c$$

- $R(X) = \lambda \sum_n \left[||(G_x * x)[n]||^2 + ||(G_x * x)[n]||^2\right]$
- $R(X) = \lambda(\|A_{gx}X\|^2 + \|A_{gy}X\|^2)$
- Using $\|Y\|^2 = Y^T Y, (AB)^T = B^T A^T$:
  - $Q = \frac{1}{2\sigma^2} I + \lambda(A_{gx}^T A_{gx} + A_{gy}^T A_{gy})$
  - $b = \frac{1}{2\sigma^2} Y$
- Q is HUGE and not diagonal.
- Can't even form Q, let alone call `scipy.linalg.solve`
- You could form 'sparse matrix', but we'll get to that later.

DENOISING

- Need to find $X = Q^{-1}b$ where
  - $Q = \frac{1}{2\sigma^2}I + \lambda(A_{gx}^T A_{gx} + A_{gy}^T A_{gy})$
  - $b = \frac{1}{2\sigma^2}Y$
- Can we diagonalize Q ?
- YES ! Use the Fourier Transform / Fourier basis S
  - $A_{gx} = SD_{gx}S^*$
  - $A_{gx}^T A_{gx} = S|D_{gx}|^2 S^*$
  - $A_{gy}^T A_{gy} = S|D_{gy}|^2 S^*$
  - $I = SS^* = SIS^*$

$|D_g|^2$ denotes $D_g^* D_g$.

DENOISING

- Need to find $X = Q^{-1}b$ where
  - $Q = \frac{1}{2\sigma^2} I + \lambda(A_{gx}^T A_{gx} + A_{gy}^T A_{gy})$
  - $b = \frac{1}{2\sigma^2} Y$

$$Q = S \underbrace{\left[\frac{1}{2\sigma^2} I + \lambda(|D_{gx}|^2 + |D_{gy}|^2) \right]}_{\text{Diagonal}} S^*$$

$$QX = b \rightarrow S^* X = \left[\frac{1}{2\sigma^2} I + \lambda(|D_{gx}|^2 + |D_{gy}|^2) \right]^{-1} S^* b$$

$$F_X[u, v] = \left[\frac{1}{2\sigma^2} + \lambda(|F_{gx}[u, v]|^2 + |F_{gy}[u, v]|^2) \right]^{-1} \frac{F_Y[u, v]}{2\sigma^2}$$

$$F_X[u, v] = \frac{F_Y[u, v]}{1 + 2\sigma^2 \lambda(|F_{gx}[u, v]|^2 + |F_{gy}[u, v]|^2)}$$

- Caveat: Assumes circular convolution

DE-BLURRING

$$X = \arg\min_X \frac{1}{2\sigma^2}\|Y - A_k X\|^2 + \lambda\left(\|A_{gx}X\|^2 + \|A_{gy}X\|^2\right)$$

$$X = \arg\min_X X^T Q X - 2b^T X + c$$

- $b = \frac{1}{2\sigma^2} A_k^T Y$
- $Q = \frac{1}{2\sigma^2} A_k^T A_k + \lambda(A_{gx}^T A_{gx} + A_{gy}^T A_{gy})$
- Still diagonalizable by the Fourier Basis

$$Q = S \underbrace{\left[\frac{1}{2\sigma^2}|D_k|^2 + \lambda(|D_{gx}|^2 + |D_{gy}|^2\right]}_{\text{Diagonal}} S^*$$

$$QX = b \rightarrow S^* X = \left[\frac{1}{2\sigma^2}|D_k|^2 + \lambda(|D_{gx}|^2 + |D_{gy}|^2\right]^{-1} S^* b$$

- $S^* A_k^T Y = S^*(S D_K S^*)^* Y = D_K^* \; S^* Y$

DE-BLURRING

$$X = \arg\min_X \frac{1}{2\sigma^2} \|Y - A_k X\|^2 + \lambda \left(\|A_{gx} X\|^2 + \|A_{gy} X\|^2 \right)$$

$$X = \arg\min_X X^T Q X - 2b^T X + c$$

$$F_X[u,v] = \frac{\bar{F}_k[u,v] F_Y[u,v]}{|F_k[u,v]|^2 + 2\sigma^2 \lambda (|F_{gx}[u,v]|^2 + |F_{gy}[u,v]|^2)}$$

- When $\lambda = 0$, $F_X = F_Y / F_k$.
- But this is unstable since $F_k[u,v]$ can be 0 for some $[u,v]$.
- We can see that the regularization term in the denominator dominates for u, v where $|F_k[u,v]|^2$ is low.
- This is called Wiener filtering.
- Again remember, assumes circular convolution.

GENERIC RESTORATION

$$X = \arg\min_X \sum_n w[n] \, \|Y[n] - (X * k)[n]\|^2 \quad + \quad R(x)$$

$$X = \arg\min_X \|D_{\sqrt{w}}(Y - A_k X)|^2 + R(x)$$

$$X = \arg\min_X \quad X^T (A_k^T D_w A_k) X \quad - \quad 2 A_k^T D_w Y \quad + \quad R(x)$$

$$X = \arg\min_X X^T Q X - 2 b^T X + c$$

- Now, Q is no longer diagonalized by the Fourier Basis
- No other choice but Cholesky ?
- Q is hard to form, but we can compute $Q\ v$ for any v very easily.
  $Q\ v = A_k^T D_w A_k + \lambda \left(A_{gx}^T A_{gx} + A_{gy}^T A_{gy} \right)$
  - This takes an "image" shaped vector and returns an image shaped vector.
  - Multiplication by A_k, A_{gx}, A_{gy} is convolution by corresponding kernels.
  - Multiply by D_w is a point-wise operation.
  - Multiply by A_k^T is convolution with flipped kernel.

CONJUGATE GRADIENT

- Generic algorithm for solving $Qx = b$ for symmetric positive definite Q.
- Useful when you can multiply by Q but not 'form' it.

Basic Idea

- For a given set of vectors $\{p_1, p_2, \ldots p_N\}$
  - that are same size as x
  - linearly independent
  - N = dimensionality of x
- We can write any $x = \sum_i \alpha_i p_i$
- If we also choose the vectors to be 'conjugate' such that $p_i^T Q p_j = 0$ for $i \neq j$:

$$Qx = b \rightarrow p_k^T Qx = p_k^T b \rightarrow \alpha_i p_k^T Q p_k = p_k^T b \rightarrow \alpha_i = \frac{p_k^T b}{p_k^T Q p_k}$$

CONJUGATE GRADIENT

Iterative Algorithm

- Begin with some guess x_0 for x (say all zeros)
- $k = 0, r_0 \leftarrow b - Qx_0, \quad p_0 \leftarrow r_0$
- Repeat
  - $\alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T Q p_k}$
  - $x_{k+1} = x_k + \alpha_k p_k$
  - $r_{k+1} = r_k - \alpha_k Q p_k$
  - $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
  - $p_{k+1} = r_{k+1} + \beta_k p_k$
  - $k = k + 1$

Stop at some measure of convergence. Pre-conditioned variants. Additional reading:
https://en.wikipedia.org/wiki/Conjugate_gradient_method

DE-BLURRING

What if we did not have a squared regularizer on gradients ?

$$X = \arg\min_X \sum_n \|Y[n] - (X * k)[n]\|^2 + \lambda \sum_n \left(\|(G_x * X)[n]\| + \|(G_y * X)[n]\|\right)$$

No longer a quadratic form. ($\|\cdot\|$ implies absolute value)

Variable splitting (Divide and Concur) Approach

$$X = \arg\min_X \min_{\{c_x[n], c_y[n]\}} \sum_n \|Y[n] - (X * k)[n]\|^2 + \lambda \sum_n \left(\|c_x[n]\| + \|c_y[n]\|\right)$$

$$+ \beta \left[\sum_n ((G_x * X)[n] - c_x[n])^2 + \left((G_y * X)[n] - c_y[n]\right)^2\right]$$

Equivalent when $\beta \to \infty$

DE-BLURRING

$$X = \arg\min_X \min_{\{c_x[n], c_y[n]\}} \sum_n \|Y[n] - (X * k)[n]\|^2 + \lambda \sum_n \left(\|c_x[n]\| + \|c_y[n]\|\right)$$

$$+ \beta \left[\sum_n ((G_x * X)[n] - c_x[n])^2 + \left((G_y * X)[n] - c_y[n]\right)^2\right]$$

Iterative Approach

- Begin with some estimate of X, and a small value of β
- Alternate between
  - Minimizing wrt c_x, c_y keeping X constant. Pointwise.
  - Minimizing wrt X keeping c_x, c_y constant. Quadratic / Fourier diagonalized.
  - While increasing the value of β

Further Reading: Krishnan and Fergus. Fast Image Deconvolution using Hyper-Laplacian Priors, NIPS 2009. Also see the ADMM algorithm.

COLOR

COLOR

Remember, at each pixel:

$$X_r[n] = \int_\lambda L(\lambda, n)\Pi_r(\lambda)d\lambda$$
$$X_g[n] = \int_\lambda L(\lambda, n)\Pi_g(\lambda)d\lambda$$
$$X_b[n] = \int_\lambda L(\lambda, n)\Pi_b(\lambda)d\lambda$$

- $L(\lambda, n)$ is the light incident at n
  - We've folded in spatial sensitivity, quantum efficiency, ignored noise.
- Here Π_r, Π_g, Π_b are the wavelength-dependent transmissions of the camera's color filters.
  - Often called color matching functions.
- Assume these are RAW images (no post-processing).

COLOR

Remember, at each pixel:

$$X_r[n] = \int_\lambda L(\lambda, n)\Pi_r(\lambda)d\lambda$$
$$X_g[n] = \int_\lambda L(\lambda, n)\Pi_g(\lambda)d\lambda$$
$$X_b[n] = \int_\lambda L(\lambda, n)\Pi_b(\lambda)d\lambda$$

Observations

- This is "projection" of a continuous valued function to three numbers.
  - Loss of information.
  - Metamerism: $L(\lambda)$ that have the same RGB values.

COLOR

Remember, at each pixel:

$$X_r[n] = \int_\lambda L(\lambda, n)\Pi_r(\lambda)d\lambda$$
$$X_g[n] = \int_\lambda L(\lambda, n)\Pi_g(\lambda)d\lambda$$
$$X_b[n] = \int_\lambda L(\lambda, n)\Pi_b(\lambda)d\lambda$$

Observations

- Rationale: Models the human visual system.
    - We only have three kind of photoreceptors
    - The standard R,G,B filters "span" the same subspace as human observers.
        - Determined using psycho-physical experiments
        - By the International Commission on Illumination (CIE) in 1931
        - Introduced the concept of primary colors
        - Defined the CIE standard observer

We can't distinguish between metamers either.

COLOR

Remember, at each pixel:

$$X_r[n] = \int_\lambda L(\lambda, n)\Pi_r(\lambda)d\lambda$$
$$X_g[n] = \int_\lambda L(\lambda, n)\Pi_g(\lambda)d\lambda$$
$$X_b[n] = \int_\lambda L(\lambda, n)\Pi_b(\lambda)d\lambda$$

Observations

- $L(\lambda)$ is the spectrum of the light that reaches the camera.
  - This is a function of both the object surface, and the illumination
  - Lights can be of different colors
  - But human perception of color is very stable under changing illumination
  - **"Color Constancy"**
- Also means metamerism is illumination dependent
  Two objects could have identical RGB values under one light but not another.